

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

| | | | |
|---|--|---|--------------------------------------|
| 1. AGENCY USE ONLY (Leave blank) | 2. REPORT DATE 9/20/96 | 3. REPORT TYPE AND DATES COVERED Final Report | |
| 4. TITLE AND SUBTITLE Translation from More to Less Expressive Representation Languages | | 5. FUNDING NUMBERS F49620-92-J-0434 | |
| 6. AUTHOR(S) Jeffrey Van Baalen | | | |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) University of Wyoming Computer Science Department | | AFOSR-TR-96 C492 | |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Wendy M. Veon AFORS/PKA 110 DUNCAN AVENUE SUITE B 115 BOLLING AFB DC 20332-0001 | | 10. SPONSORING/MONITORING AGENCY REPORT NUMBER NM | |
| 11. SUPPLEMENTARY NOTES | | | |
| 12a. DISTRIBUTION/AVAILABILITY STATEMENT UL 19961017 135 | | 12b. DISTRIBUTION CODE | |
| 13. ABSTRACT (Maximum 200 w) Knowledge Base reuse is a real problem of great practical significance. Currently it is not possible to reuse a knowledge base developed for another purpose. As a result, knowledge-based systems are very expensive to develop because it is always necessary to start from scratch on a new knowledge base. A significant stumbling block to knowledge base reuse is that all practically usable knowledge bases are developed in specialized representation languages. Hence, reuse requires translation between specialized languages. This type of translation is particularly difficult when the target language is less expressive than the source. The goal of this project is to develop a translator shell that enables the rapid development of translators between specialized representation languages. In the first year, a prototype of the shell was developed and feasibility demonstrated with the development of a Knowledge Interchange Format (KIF) to CLASSIC translator. In the second year, an EXPRESS to KIF translator and a LOOM to KIF translator were completed and tested on several example knowledge bases, the translator shell was significantly enhanced to support several new capabilities, experiments were successfully conducted to show that Ontolingua could be significantly improved by replacing its translators with translators developed using our shell, an activity to apply our work to translation between databases was begun. In the third year, a prototype translator between human genome databases was developed, support of the translator shell technology continued, and further research was conducted on the KIF to KIF translation problem | | | |
| DTIC QUALITY INSPECTED 1 | | | |
| 14. SUBJECT TERMS Knowledge representation, formal language translation | | 15. NUMBER OF PAGES 7 | |
| | | 16. PRICE CODE | |
| 17. SECURITY CLASSIFICATION OF REPORT unclassified | 18. SECURITY CLASSIFICATION OF THIS PAGE unclassified | 19. SECURITY CLASSIFICATION OF ABSTRACT unclassified | 20. LIMITATION OF ABSTRACT UL |

Final Report
AFOSR #F49620-92-J-0434
PI: Jeffrey Van Baalen
University of Wyoming

Abstract

Knowledge Base reuse is a real problem of great practical significance. Currently it is not possible to reuse a knowledge base developed for another purpose. As a result, knowledge-based systems are very expensive to develop because it is always necessary to start from scratch on a new knowledge base. A significant stumbling block to knowledge base reuse is that all practically usable knowledge bases are developed in specialized representation languages. Hence, reuse requires translation between specialized languages. This type of translation is particularly difficult when the target language is less expressive than the source. The goal of this project is to develop a translator shell that enables the rapid development of translators between specialized representation languages. In the first year, a prototype of the shell was developed and feasibility demonstrated with the development of a Knowledge Interchange Format (KIF) to CLASSIC translator. In the second year, an EXPRESS to KIF translator and a LOOM to KIF translator were completed and tested on several example knowledge bases, the translator shell was significantly enhanced to support several new capabilities, experiments were successfully conducted to show that Ontolingua could be significantly improved by replacing its translators with translators developed using our shell, an activity to apply our work to translation between databases was begun. In the third year, a prototype translator between human genome databases was developed, support of the translator shell technology continued, and further research was conducted on the KIF to KIF translation problem.

1. Results

In the first phase of this effort, a KIF-CLASSIC translator was completed and a series of tests were performed. The culmination of these tests translated the ROME Planning Initiative knowledge base from LOOM to KIF using a LOOM-KIF translator developed by Ramesh Patil at USC ISI and then the KIF-CLASSIC translator was used to translate the result into CLASSIC. One would not expect the translation from KIF-CLASSIC to be 100% successful since LOOM is a strictly more expressive language than CLASSIC. In fact, approximately 80% of the KIF version of this knowledge base was translated into CLASSIC. Subsequent analysis showed that there exists no translation into CLASSIC for the remaining 20% of the KIF knowledge base.

Hence, the KIF-CLASSIC translator succeeded in translating a real LOOM knowledge base into CLASSIC. Every KIF statement generated by the LOOM-KIF translator that was representable in CLASSIC was translated by the KIF-CLASSIC translator. The KIF-CLASSIC translator's ability to flag untranslatable statements proved useful in several ways including debugging the LOOM-KIF translator.

Two conclusions were apparent from the experiences with this first translator: more general techniques would be necessary for translation from more to less expressive languages in general, but much of the technology that we had developed could be used to

develop less powerful translators that translated between equally as expressive subsets of two languages.

In phase two, employing the above observation, a LOOM-KIF and an EXPRESS-KIF translator were developed. These translators were tested on equally expressive subsets of the respective languages with 100% success. For example, the EXPRESS-KIF translator was tested by translating several parts of the PDES/STEP knowledge base into KIF and then back into EXPRESS.

The KIF-CLASSIC, LOOM-KIF and EXPRESS-KIF translators have been made available to the research community.

Two other efforts were pursued in parallel to building the LOOM-KIF and EXPRESS-KIF translators: application of the technology already developed for translation between representation languages for database interoperability and development of more general techniques for KIF to KIF translation.

On the first of these, we completed a prototype of a human genome database translator. This proved to be a straightforward use of our translator shell, except that a genetics Ontology needed to be constructed to use as an interlingua. The prototype translates the two human genome database languages GenBank and Swiss Protégé [Van Baalen & Looby 95]. Several organizations involved in the Human Genome project have expressed interest in using the translator.

In our work to develop more general techniques for KIF to KIF translation, we extended our DCTGs to translate languages that are not context free. Many representation languages in which we are interested have this property, i.e., they are not (unordered) sets of independent top-level forms. Instead, the presence of certain top-level forms can effect the translations of others. In addition, in some languages there are ordering constraints on top-level forms so that the meaning of forms occurring in one order is different from their meaning when they occur in a different order. The user's model of our extensions to DCTGs are briefly described here. More detail can be found in [Van Baalen, et al. 95].

LOOM is an example of a language in which the presence of some top-level forms effects the translation of others. For instance, LOOM has the notion of a partitioning of a concept. This is a set of subconcepts of the concept that are all pairwise disjoint; e.g., dog and person partition animal. To illustrate the need for this capability, consider the LOOM excerpt below (which has been simplified for presentation purposes).

```
(defconcept biblio-thing
  :partitions $biblio-thing-partition-1$
  :annotations ((documentation "Biblio-thing is the root of the bibliographic
ontology."))
  :is-primitive INDIVIDUAL)

(defconcept biblio-text
  :in-partition $biblio-thing-partition-1$
  :annotations ((documentation "The most general class of undifferentiated text
objects."))
  :is-primitive (:and biblio-thing string))

(defconcept agent
  :in-partition $biblio-thing-partition-1$
```

:annotations ((documentation "An agent is something that can act on its own and produce changes in the world. There is more to agenthood than that, but for this ontology that is all that matters.")
:is-primitive biblio-thing)

There are three top-level forms defining the concepts **biblio-thing**, **biblio-text**, and **agent**. **Biblio-thing** is the root concept of a bibliographic ontology. The concepts **agent** and **biblio-text** partition **biblio-thing**, a fact that is expressed in KIF as:

(subclass-partition biblio-thing (setof agent biblio-text))

To identify this partitioning and properly translate these forms into KIF, all three top-level forms must be analyzed before a translation is generated. The KIF translation of this excerpt is:

(defrelation biblio-thing
 (class biblio-thing)
 (subclass-of biblio-thing individual)
 (subclass-partition biblio-thing (setof agent biblio-text)))
(documentation biblio-thing
 "biblio-thing is the root of the bibliographic ontology.")
(defrelation agent
 (class agent)
 (subclass-of agent biblio-thing)))
(documentation agent
 "an agent is something or someone that can act on its own and produce changes in the world. There is more to agenthood than that, but for this ontology that is all that matters.")
(documentation biblio-text
 "the most general class of undifferentiated text objects.")
(defrelation biblio-text
 (class biblio-text)
 (subclass-of biblio-text biblio-thing)
 (subclass-of biblio-text string))

To enable this type of translation and also translation where the order of top-level forms effects their translation, DCTGs have been extended so that one can specify the translations of sets and sequences of top-level forms. This is done using the special function symbols **set** and **seq** in the first and second arguments to the top-level predicate **TRANS**.

Grammar rules containing **set** or **seq** forms in the first argument specify how subsets or subsequences of a knowledge base translate into the interlingua. This is a significant departure from simple DCTGs that specify translation one top-level form at a time. Translation rules that contain **set** or **seq** forms in their second argument specify that the translations generated by those rules are not single top-level forms, but rather are sets or sequences of top-level forms in the interlingua.

For example, the translation of the sequence **A, B** can be specified directly in a translation rule as follows:

(<- (trans (seq A B) B-with-A)).

To handle the translation of **B** alone as **B-without-A**, the clause

(<- (trans B B-without-A))

is placed after the above clause. The standard semantics of PROLOG ensure that the first rule takes precedence over the second.

Similarly, the function **set** can be used to specify that the order in which top-level forms appear is unimportant, e.g.,

```
(<- (trans (set A B) B-with-A))
```

will translate the two top-level forms as **B-with-A** regardless of the order in which they appear.

The **set** and **seq** function symbols can be nested so that, for example, one can specify with the term **(set A (seq B C))** that **C** must occur after **B**, but **A** can appear in any relationship to **B** and **C**. An extended DCTG for translating the example LOOM knowledge base given above is:

```
(defrule trans
  ; Translate a defconcept that has a :partitions field.
  (<- (trans (set (defconcept ?C
                    :partitions ?p-name
                    :annotations ((documentation ?document))
                    . ?sent-rest)
                    . ?kb-rest)
              (set (documentation ?C ?document)
                  (defrelation ?C
                    (subclass-partition ?C . ?p-members)
                    . ?new-sent-rest)
                    . ?new-kb-rest))
              (partition-name ?p-name)
              (translate2 (defconcept ?C . ?sent-rest)
                  (defrelation ?C . ?new-sent-rest))
              (trans2 ?kb-rest ?new-kb-rest ?p-name ?p-members))

  ; Translate defconcepts that have neither a :partitions
  ; field nor an :in-partitions field
  (<- (trans (defconcept ?C . ?sent-rest) (defrelation ?C . ?new-sent-rest))
      (not-member-p :in-partition ?sent-rest)
      (not-member-p :partitions ?sent-rest)
      (translate2 (defconcept ?C . ?sent-rest) (defrelation ?C . ?new-sent-rest)))

  ; Translate top-level forms that are not defconcept forms
  (<- (trans (?a . ?rest) ?all-trans)
      (not-equal-p ?a defconcept)
      (translate (?a . ?rest) ?all-trans)))

(defrule trans2
  ; Translate defconcepts for concepts that are in the
  ; partition ?p-name (passed in by trans).
  (<- (trans2 (set (defconcept ?c
                    :in-partition ?p-name
                    :annotations ((documentation ?doc))
                    . ?sent-rest)
                    . ?kb-rest)
              (set (documentation ?c ?doc)
                  (defrelation ?C
```

```

(subclass-partition ?C . ?p-members2)
. ?new-sent-rest)
. ?new-kb-rest)
?p-name
(?C . ?p-member-rest))
(translate2 (defconcept ?C . ?sent-rest) (defrelation ?C . ?new-sent-rest))
(trans2 ?kb-rest ?new-kb-rest ?p-name ?p-member-rest))
(<- (trans2 nil nil ?p-name nil)))

```

The predicate **trans2**, used in the above DCTG but not defined here, is responsible for translating individual LOOM top-level forms.

We also began formalizing our approach to translation between subsets of KIF. The approach we have taken to constructing procedures for this type of translation is inspired by techniques that have been developed for term rewriting. Research in this area studies the problem of reasoning efficiently about the consequences of equational theories. One basic contribution of this work is a technique for developing decision procedures for this consequence problem which is undecidable in general. That is, let T be an equational theory and t_1 and t_2 be terms in the language of T . Then it is undecidable whether or not $t_1 =_T t_2$, i.e., $T \vdash (t_1 = t_2)$. However, sometimes the problem of determining $t_1 =_T t_2$ can be reduced using rewrite systems. Specifically, it is sometimes possible to find a system of rewrite rules for T such that $t_1 =_T t_2$ just in case t_1 and t_2 are rewritten to identical terms. Such a rewrite system constitutes a decision procedure for the relation $=_T$. When such a system can be found, the term to which t_1 and t_2 are rewritten is called their *canonical* form.

We are developing a similar idea for translating between subsets of KIF. Before we describe our approach, we make some important definitions. First, we make precise the notion of a syntactically restricted subset of a language.

Definition (syntactically restricted subset) Let L_i be a language and let G be a definite clause grammar (DCG) that parses a strict subset of L_i . The subset of L_i that G parses is called a *syntactically restricted subset of L_i* .

Note that a DCG for a syntactically restricted subset of KIF can be extracted from a DCTG that translates between a representation language and KIF. Now we define the idea of a congruence relation between two syntactically restricted subsets of a language.

Definition (congruence relation on subsets) Let A and B be two syntactically restricted subsets of a language L_i and let R be a binary relation whose domain is a subset of A and whose range is a subset of B . R is a *congruence relation between A and B* in the following case:

$$\forall x, y, x', y' [(R(x, y) \wedge (x \leftrightarrow x') \wedge R(x', y')) \Rightarrow (y \leftrightarrow y')]$$

Hence, if a top-level form x in subset A is related to $y \in B$, and a top-level form $x' \in A$ is equivalent to x and related to y' , then for R to be a congruence relation on A and B it must be the case that y' is equivalent to y .

Finally, we define a canonical form for a congruence relation between two syntactically restricted subsets of a language.

Definition (canonical form for a congruence between subsets) Let L_i be a language, A and B be syntactically restricted subsets of L_i and C also be a subset of L_i . Let R be a congruence relation between A and B and let c be a mapping from $A \cup B$ into C . C is a

canonical form for R and c is a *canonical form mapping for R* just in case $R(x,y) \Leftrightarrow \text{identical}(c(x),c(y))$.

Hence, when c is a canonical form mapping for R , x and y are related by R just in case their canonical forms are identical.

We can now describe our approach to developing translators between subsets of KIF. We believe that the automatic derivation of canonical forms, as in done in the rewrite community, is beyond the current state of the technology. Therefore, our approach is to provide tools that assist a translator developer with the construction of:

- a *canonical form* for a congruence relation between two syntactically restricted subsets of KIF
- a DCTG for translating KIF top-level forms into this canonical form (this is an implementation of c) and
- modified versions of the source and target language DCTGs that translate the canonical form of KIF to and from the specialized representation languages.

The translator developer begins with a DCTG that translates between the source language and the source subset of KIF and a DCTG that translates between the target language and the target subset of KIF. He or she then specifies a congruence relation between the source and target subsets of KIF. This specification is a set of equivalences between top-level forms in the source subset and top-level forms in the target subset and equalities between first-order terms in the source subset and first-order terms in the target subset.

Then the developer attempts to construct a canonical form for this congruence relation and a DCTG that translates both subsets to this form. The translator shell assists in this process in several ways. First, it provides a library of "standard" canonical forms and associated DCTGs. An example of such a form is conjunctive normal form. The developer can use, modify, or, in some cases, combine these forms and their associated DCTGs. Second, the shell provides tools for proving *termination*, i.e., that a DCTG always terminates, and *coverage*, i.e., that every top-level form in the target and source subsets of KIF are translated into the proposed canonical form. The developer uses these tools to prove termination and coverage of new or modified DCTGs.

Finally, the shell is used to establish that the proposed canonical form is, in fact, canonical for the congruence relation. Explaining how this is done requires one more definition.

Definition (canonical for an equivalence) Let c be a mapping from formulas in a language L_i into C , a subset of L_i and let $\phi, \psi \in L_i$. If $\text{identical}(c(\phi), c(\psi))$, c is called a *canonical form mapping for Γ* where Γ is the universal closure of $\phi \Leftrightarrow \psi$. Also, C is called a *canonical form for Γ* .

A *canonical form for an equality* is defined similarly. We have shown that if S is a specification of a congruence relation between two subsets of a language and c is a canonical form mapping for every equivalence and every equality in S , then c is a canonical form mapping for the congruence relation specified by S . Hence, the translator shell establishes that a form is canonical for a congruence relation by using the associated DCTG to translate the left and right hand sides of every formula in the specification of that relation and checking that their translated forms are identical.

Once a canonical form for a congruence relation between two subsets of KIF has been constructed, the translator developer can return to the original two DCTGs and modify

them so that they translate into and out of the canonical form. We propose to explore methods of assisting the developer in this process (see section 4.4 for more detail).

2. Conclusions

The author believes that significant initial success was achieved under this grant in addressing the problem of translation between representation languages. However, the knowledge representation community has been slow in showing interest in using our results. On the other hand, our achievements in database interoperability in the experiment to translate between human genome databases have led to an investigation of the use of this technology in solving interoperability problems in spatial and geographical databases. This work will continue to be funded by a National Science Foundation Epscore grant to the University of Wyoming.

3. List of Publications

Fikes, R.E., Cutkosky, M., Gruber, T., and Van Baalen, J., 1991, "Knowledge Sharing Technology Project Overview," Stanford University Report KSL 91-71.

Deng, Y., "An EXPRESS to KIF translator," University of Wyoming Masters Thesis, 1994.

Van Baalen, J., "Knowledge engineering from sets of examples," Proceedings of the 1994 workshop on change of representation and problem reformulation, Jackson Hole Wyoming, 1994.

Van Baalen, J. and Fikes, R. E., "The role of reversible grammars in translation between knowledge representation languages," Proceedings of the International Conference on Knowledge Representation and Reasoning, Bonn, 1994.

Lowry, M. L. & Van Baalen, J., "Meta-Amphion: Synthesis of Efficient Domain Specific Synthesis Systems," to appear in the proceedings of the Conference on Knowledge-Based Software Engineering, 1995.

Van Baalen, J. & Looby, J. 1995. "Translation between Human Genome Databases," Proceedings of the IJCAI95 Workshop on Basic Ontological Issues in Knowledge Sharing.

Van Baalen J., & Lowry, M.L., 1995, "A New Generation Deductive Synthesis System," Proceedings of the 1995 Symposium on Abstraction Reformulation and Approximation.